UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/828,794	04/21/2004	Erik Altman	YOR090040015US1 (163-29)	5081

24336 7590 04/17/2007

KEUSEY, TUTUNJIAN & BITETTO, P.C.
20 CROSSWAYS PARK NORTH
SUITE 210
WOODBURY, NY 11797

EXAMINER	
FENNEMA, ROBERT E	
ART UNIT	**PAPER NUMBER**
2183	

SHORTENED STATUTORY PERIOD OF RESPONSE	MAIL DATE	DELIVERY MODE
3 MONTHS	04/17/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

If NO period for reply is specified above, the maximum statutory period will apply and will expire 6 MONTHS from the mailing date of this communication.

Office Action Summary	Application No.	Applicant(s)	
	10/828,794	ALTMAN ET AL.	
	Examiner	Art Unit	
	Robert E. Fennema	2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *12 January 2007*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-30* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-30* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-30 have been considered. Claims 1 and 22 have been amended as per Applicant's request.

Claim Rejections - 35 USC § 102

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

2. Claims 1-30 are rejected under 35 U.S.C. 102(b) as being anticipated by Tran (USPN 5,765,035).

3. As per Claim 1, Tran teaches: A pipeline, comprising:

a plurality of operational stages (Abstract), the stages including:

a pointer register stage which stores pointer information and updates (Column 9, Lines 32-37, the reservation stations. The stations hold instruction information to be executed by the functional units. Column 7, Lines 31-57 disclose that registers, which generally hold the information in the reservation stations, can make use of indirect addressing, such that the value of the register is used as an address (or pointer) to a location in memory, where the real data is found. In this embodiment, the basic x86 registers function as "pointers", which can be stored in the reservation station prior to execution);

a pointer dependency checking stage located downstream of the pointer register stage, which determines if instruction pointer dependencies exist and stalls an instruction prior to issuance if necessary to resolve inter-instruction dependencies

(Column 7, Lines 60-64, the reorder buffer does dependency checking, and as seen by Column 9, Lines 51-64, the reorder buffer "tags" an entry in a reservation station if an operand is not available, causing the instruction to stall, being unable to issue until the tag is replaced by a value. Column 11, Line 54 – Column 12, Line 17 deals with stalling and dependency checking in the memory operation case);

at least one functional unit providing pointer information updates to the pointer register stage (Column 9, Lines 45-50) such that the pointer information is processed and updated to the pointer register stage before an instruction goes through the dependency checking stage, an issue stage, and an execution stage (Column 9, Lines 45-60).

4. As per Claim 2, Tran teaches: The pipeline as recited in claim 1, further comprising a pointer execution stage used before the dependency checking stage such that inter-instruction dependency is checked after the pointer execution stage or in parallel with pointer execution (Column 9, Lines 45-50. The functional units are part of the pointer execution stage).

5. As per Claim 3, Tran teaches: The pipeline as recited in claim 2, further comprising a path for making pointer updates available to the pointer register stage before the instruction reaches a write back stage of the pipeline (Column 9, Lines 45-50. The result is bypassed back to the reservation stations (pointer register stage), as the

same time it goes to the reorder buffer).

6. As per Claim 4, Tran teaches: The pipeline as recited in claim 3, wherein the path includes a normal pointer update path which returns pointer information from the at least one functional unit (Column 9, Lines 45-50).

7. As per Claim 5, Tran teaches: The pipeline as recited in claim 3, further comprising a pointer execution bypass making pointer updates available to immediately following instructions before a pointer update is written into the pointer register file (Column 9, Lines 45-50).

8. As per Claim 6, Tran teaches: The pipeline as recited in claim 2, further comprising a pointer reorder buffer coupled to the pointer register stage to maintain a precise state of pointers (Column 7, Lines 60-64, it is part of the dependency checking stage).

9. As per Claim 7, Tran teaches: The pipeline as recited in claim 6, further comprising a precise pointer file for storing the precise state of the pointer reorder buffer (Figure 1, Register File 218).

10. As per Claim 8, Tran teaches: The pipeline as recited in claim 7, further comprising an interrupt recovery path which keeps the pointer register stage up to date

with reordering or recovery information from the precise pointer file (Column 10, Lines 6-16).

11. As per Claim 9, Tran teaches: The pipeline as recited in claim 1, further comprising a combined pointer reorder buffer/issue stage coupled to the dependence checking stage to issue instructions and maintain a precise state/order of pointers (Column 7, Lines 60-64, and Column 8, Line 46 – Column 9, Line 9. The reorder buffer maintains the ordering of the operations, and helps issue in the sense that it forwards appropriate data to the reservation station to allow it to issue).

12. As per Claim 10, Tran teaches: The pipeline as recited in claim 9, wherein the combined pointer reorder buffer/issue stage includes a table with a plurality of fields to keep identifiers of all pointers that are updated by an instruction, and new values of the updated pointers (Column 8, Line 46 – Column 9, Line 6).

13. As per Claim 11, Tran teaches: The pipeline as recited in claim 9, further comprising a precise pointer file for storing the precise state of the combined pointer reorder buffer/issue stage (Figure 1, Register File 218).

14. As per Claim 12, Tran teaches: The pipeline as recited in claim 11, further comprising an interrupt recovery path, which restores the pointer register stage to the precise state from the precise pointer file (Column 10, Lines 6-16).

15. As per Claim 13, Tran teaches: A pipeline, comprising:

a plurality of operational stages (Abstract), the stages including:

a pointer register stage which stores pointer information and updates (Column 9, Lines 32-37, the reservation stations. The stations hold instruction information to be executed by the functional units. Column 7, Lines 31-57 disclose that registers, which generally hold the information in the reservation stations, can make use of indirect addressing, such that the value of the register is used as an address (or pointer) to a location in memory, where the real data is found. In this embodiment, the basic x86 registers function as "pointers", which can be stored in the reservation station prior to execution);

a dependence checking stage located downstream of the pointer register stage, which determines if instruction dependencies exist and stalls an issue prior to issuance if necessary to resolve inter-instruction dependencies (Column 7, Lines 60-64, the reorder buffer does dependency checking, and as seen by Column 9, Lines 51-64, the reorder buffer "tags" an entry in a reservation station if an operand is not available, causing the instruction to stall, being unable to issue until the tag is replaced by a value. Column 11, Line 54 – Column 12, Line 17 deals with stalling and dependency checking in the memory operation case);

a pointer execution stage for processing pointers prior to the dependence checking stage (Column 9, Lines 45-50. The functional units are part of the pointer execution stage), the pointer execution stage providing pointer updates to the pointer

register stage via an early pointer update path (Column 9, Lines 45-50. The result is bypassed back to the reservation stations (pointer register stage), as the same time it goes to the reorder buffer); and

at least one functional unit providing pointer information updates to the pointer register stage such that pointer information is processed and updated to the pointer register stage (Column 9, Lines 45-50).

16. As per Claim 14, Tran teaches: The pipeline as recited in claim 13, further comprising a pointer reorder buffer coupled to the pointer register stage to maintain a precise state of pointers (Column 7, Lines 60-64, it is part of the dependency checking stage).

17. As per Claim 15, Tran teaches: The pipeline as recited in claim 14, further comprising a precise pointer file for storing the precise state of the pointer reorder buffer (Figure 1, Register File 218).

18. As per Claim 16, Tran teaches: The pipeline as recited in claim 15, further comprising an interrupt recovery path which keeps the pointer register stage up to date with reordering or recovery information from the precise pointer file (Column 10, Lines 6-16).

19. As per Claim 17, Tran teaches: The pipeline as recited in claim 13, further comprising a normal pointer update path which returns pointer information from the at least one functional unit (Column 9, Lines 45-50).

20. As per Claim 18, Tran teaches: The pipeline as recited in claim 13, further comprising a combined pointer reorder buffer/issue stage coupled to the dependence checking stage to issue instructions and maintain a precise state/order of pointers (Column 7, Lines 60-64, and Column 8, Line 46 – Column 9, Line 9. The reorder buffer maintains the ordering of the operations, and helps issue in the sense that it forwards appropriate data to the reservation station to allow it to issue).

21. As per Claim 19, Tran teaches: The pipeline as recited in claim 18, wherein the combined pointer reorder buffer/issue stage includes a table with a plurality of fields to keep identifiers of all pointers that are updated by an instruction, and new values of the updated pointers (Column 8, Line 46 – Column 9, Line 6).

22. As per Claim 20, Tran teaches: The pipeline as recited in claim 18, further comprising a precise pointer file for storing the precise state of the combined pointer reorder buffer/issue stage (Figure 1, Register File 218).

23. As per Claim 21, Tran teaches: The pipeline as recited in claim 20, further comprising an interrupt recovery path, which restores the pointer register stage to the

precise state from the precise pointer file (Column 10, Lines 6-16).

24. As per Claim 22, Tran teaches: A method for updating pointers ahead of an instruction, comprising the steps of:

providing a plurality of operational stages (Abstract), including a pointer register stage which stores pointer information and updates (Column 9, Lines 32-37, the reservation stations. The stations hold instruction information to be executed by the functional units. Column 7, Lines 31-57 disclose that registers, which generally hold the information in the reservation stations, can make use of indirect addressing, such that the value of the register is used as an address (or pointer) to a location in memory, where the real data is found. In this embodiment, the basic x86 registers function as "pointers", which can be stored in the reservation station prior to execution), a pointer dependence checking stage located downstream of the pointer register stage, which determines if instruction dependencies exist and stalls an issue prior to issuance if necessary to resolve inter-instruction dependencies (Column 7, Lines 60-64, the reorder buffer does dependency checking, and as seen by Column 9, Lines 51-64, the reorder buffer "tags" an entry in a reservation station if an operand is not available, causing the instruction to stall, being unable to issue until the tag is replaced by a value. Column 11, Line 54 – Column 12, Line 17 deals with stalling and dependency checking in the memory operation case), and at least one functional unit providing pointer information updates to the pointer register stage (Column 9, Lines 45-50); and

processing pointer information to update the pointer information for the pointer register stage so that updated pointer information is available (Column 8, Lines 60-65) such that the pointer information is processed and updated to the pointer register stage before an instruction goes through the dependency checking stage, an issue stage, and an execution stage (Column 9, Lines 45-60).

25. As per Claim 23, Tran teaches: The method as recited in claim 22, further comprising a step of providing pointer updates to the pointer register stage via an early pointer update path by providing a pointer execution stage used before the pointer register stage and the dependence checking stage (Column 9, Lines 45-50. The result is bypassed back to the reservation stations (pointer register stage), as the same time it goes to the reorder buffer).

26. As per Claim 24, Tran teaches: The method as recited in claim 23, further comprising a step of maintaining a precise state of pointers by employing a pointer reorder buffer (Column 7, Lines 60-64, it is part of the dependency checking stage).

27. As per Claim 25, Tran teaches: The method as recited in claim 24, further comprising a step of storing the precise state of the pointer reorder buffer in a precise pointer file (Figure 1, Register File 218 holds the precise state).

28. As per Claim 26, Tran teaches: The method as recited in claim 24, further comprising updating reordering or recovery information from the precise pointer file using an interrupt recovery path to the pointer register stage (Column 10, Lines 6-16).

29. As per Claim 27, Tran teaches: The method as recited in claim 23, further comprising maintaining a precise state/order of pointers using a combined pointer reorder buffer/issue stage coupled to the rename and dependence checking stage (Column 7, Lines 60-64, and Column 8, Line 46 – Column 9, Line 9. The reorder buffer maintains the ordering of the operations, and helps issue in the sense that it forwards appropriate data to the reservation station to allow it to issue).

30. As per Claim 28, Tran teaches: The method as recited in claim 27, wherein the combined pointer reorder buffer/issue stage includes a table with a plurality of fields to keep identifiers of all pointers that are updated by an instruction, and new values of the updated pointers (Column 8, Line 46 – Column 9, Line 6).

31. As per Claim 29, Tran teaches: The method as recited in claim 27, further comprising storing the precise state of the combined pointer reorder buffer/issue stage using a precise pointer file (Figure 1, Register File 218).

32. As per Claim 30, Tran teaches: The method as recited in claim 29, further comprising an interrupt recovery path which keeps the pointer register stage up to date

with reordering or recovery information from the precise pointer file (Column 10, Lines 6-16).

Response to Arguments

33. Applicant's arguments filed 1/12/2007 (RCE filed 1/12/2007, claims and remarks filed 11/20/2006 in an After-final Amendment) have been fully considered but they are not persuasive. Regarding Claim 1, Applicant has firstly argued Tran does not teach a pointer register stage, and argues that Examiners disclosure of Tran teaching indirect addressing being used and what indirect addressing is does not make the reservation stations a pointer register stage, and has made arguments reasoning why it can not be considered a pointer register stage. However, the claims do not represent these features (such as a pointer requiring pointer analysis), the claims only require the pointer register stage to store pointer information and updates, and the dependency checking stage only requires a check to see if pointer dependencies exist, and the reservation stations do contain pointers as Examiner has laid out in the rejection and in previous actions, in that the reservation stations can contain operands which utilize indirect addressing, which makes those operands pointers, and thus the work the reservation station does for dependency checking applies to those pointers as well. If Applicant has a more specific meaning of pointer, or requires additional checks to be made specifically for pointers, those limitations should be added to the claims, as Examiner does not find support for Applicant's arguments in the claims in a way that makes Tran not applicable.

Further regarding Claim 1, Applicant has argued that Tran does not teach updating the pointer register stage before an instruction goes through the dependency checking stage, an issue stage, and an execution stage. First of all, this limitation is broad to the point that any system with the structure as disclosed previously in the claim would read on it, as the wording is for "an instruction", which could be an instruction executed hundreds of instructions after any update, thus the fact that there is an update before "an instruction" goes through said stages is not novel as any instruction far in the future can be chosen to be the instruction. However, assuming for the sake of argument that Applicant intended for "an instruction" to represent the instruction which requires the update, which Examiner believes Applicant intended to claim, Tran does in fact teach this limitation, as Tran does teach checking dependencies before issuance (at odds in this case appears to be the interpretation of "issuance"). As stated in the Advisory action, and repeated here: "Examiner cites both Tran, Column 9, Lines 57-60, which states that instructions from the reservation station are issued to the functional units. Additionally, Examiner cites Patterson, Page 185, which states that reservation stations buffer operands of instructions "waiting to issue". Despite Tran indicating that a decode/issue unit is in place preceding a reservation station, it is not believed that the dispatching from the decode/issue unit to the reservation station could be considered an "issue", as this definition of issue would stand contrary to the common meaning of issue in the art, as well as both the textbook teachings of a reservation station by Patterson and the definition of the meaning as taught by Tran, in which an instruction can not be issued without it's operands." Based on this interpretation of issuance, Tran teaches

providing information updates to an instruction prior to issuance, as it is bypassed into the reservation station, which is part of the dependency checking stage, but as it is still in it, it has not gone through the dependency checking stage at this point, and the instruction cannot be issued without valid operands, or incorrect execution will result.

Regarding Claim 13, Examiner has responded to Applicant's arguments in the previous Advisory Action, but will repeat the arguments here for clarity: Regarding Claim 13, Applicant has argued that Tran does not teach a pointer execution stage for processing pointers prior to the dependance checking stage and an early pointer update path. First of all, Applicant claims that Examiner has stated that the function units in Tran are the reservation units, which is not what Examiner has suggested. Examiner referred to the functional units as described in Column 9, Lines 45-50 as the execution stage, as they execute instructions. At no point has Examiner implied the functional units are the reservation stations, in fact, they are quite distinct and separate objects from the reservation station, and examiner simply noted that the functional units can be bypassed back into the reservation stations through an early update (forwarding) path. Furthermore, Applicant has argued that instructions in the reservation station have already been issued. Examiner does not agree with this interpretation of the reference and the meaning of "issue". Examiner cites both Tran, Column 9, Lines 57-60, which states that instructions from the reservation station are issued to the functional units. Additionally, Examiner cites Patterson, Page 185, which states that reservation stations buffer operands of instructions "waiting to issue". Despite Tran indicating that a decode/issue unit is in place preceding a reservation station, it is not believed that the

dispatching from the decode/issue unit to the reservation station could be considered an "issue", as this definition of issue would stand contrary to the common meaning of issue in the art, as well as both the textbook teachings of a reservation station by Patterson and the definition of the meaning as taught by Tran, in which an instruction can not be issued without it's operands.

Regarding Claim 22, the above remarks address Applicant's arguments towards Claim 22 as well, as it is similar to the other independent claims argued, and Examiner believes the above arguments in combination with the explanation in the rejections also address the arguments regarding Claims 2 and 5.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Robert E. Fennema whose telephone number is (571) 272-2748. The examiner can normally be reached on Monday-Friday, 8:45-6:15.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Eddie Chan can be reached on (571) 272-4162. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Robert E Fennema
Examiner
Art Unit 2183

RF

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100